下载APP

孔令飞｜从小白到“老鸟”：我的Go语言进阶之路

2021-12-20 孔令飞

《Tony Bai · Go语言第一课》 课程介绍 >

讲述：孔令飞
时长 26:11 大小 23.99M

你好，我是孔令飞，是极客时间《Go 语言项目开发实战》专栏和《从零构建企业级 Go 项目》图书的作者，目前在腾讯云从事分布式云方向的研发工作。

很高兴，也很感谢能够借助 Tony Bai 老师的专栏《**Tony Bai •Go 语言第一课**》给你分享我的 Go 语言进阶之路。今天这一讲没有 Go 语法知识的学习，没有高大上的理论，更多的是我个人在 Go 语言进阶过程中的一些经验、心得的分享。希望通过这些分享，能帮助到渴望在 Go 研发之路上走的更远的你。

领资料

为了方便说明如何提升 Go 研发能力，我需要先给你介绍下我认为的 Go 语言能力级别划分依据。

Go 语言能力级别划分

下面这些能力级别是根据我自己的理解划分的。这些 Go 语言能力级别没有标准的定义，各级别之间也没有明确的界限，级别之间也可能有重叠的部分。但这不妨碍我们参考这些级别大概判断自己当前所处的阶段，思考如何提升自己的 Go 研发能力。

我将 Go 语言能力由低到高划分为以下 5 个级别。

初级：已经学习完 Go 基础语法课程，能够编写一些简单 Go 代码段，或者借助于 Google/Baidu 能够编写相对复杂的 Go 代码段；这个阶段的你基本具备阅读 Go 项目代码的能力；

中级：能够独立编写完整的 Go 程序，例如功能简单的 Go 工具等等，或者借助于 Google/Baidu 能够开发一个完整、简单的 Go 项目。此外，对于项目中涉及到的其他组件，我们也要知道怎么使用 Go 语言进行交互。在这个阶段，开发者也能够二次开发一个相对复杂的 Go 项目；

高级：不仅能够熟练掌握 Go 基础语法，还能使用 Go 语言高级特性，例如 channel、interface、并发编程等，也能使用面向对象的编程思想去开发一个相对复杂的 Go 项目；

资深：熟练掌握 Go 语言编程技能与编程哲学，能够独立编写符合 Go 编程哲学的复杂项目。同时，你需要对 Go 语言生态也有比较好的掌握，具备较好的软件架构能力；

专家：精通 Go 语言及其生态，能够独立开发大型、高质量的 Go 项目，编程过程中较少依赖 Google/ 百度等搜索工具，且对 Go 语言编程有自己的理解和方法论。除此之外，还要具有优秀的软件架构能力，能够设计、并部署一套高可用、可伸缩的 Go 应用。这个级别的开发者应该是团队的技术领军人物，能够把控技术方向、攻克技术难点，解决各种疑难杂症。

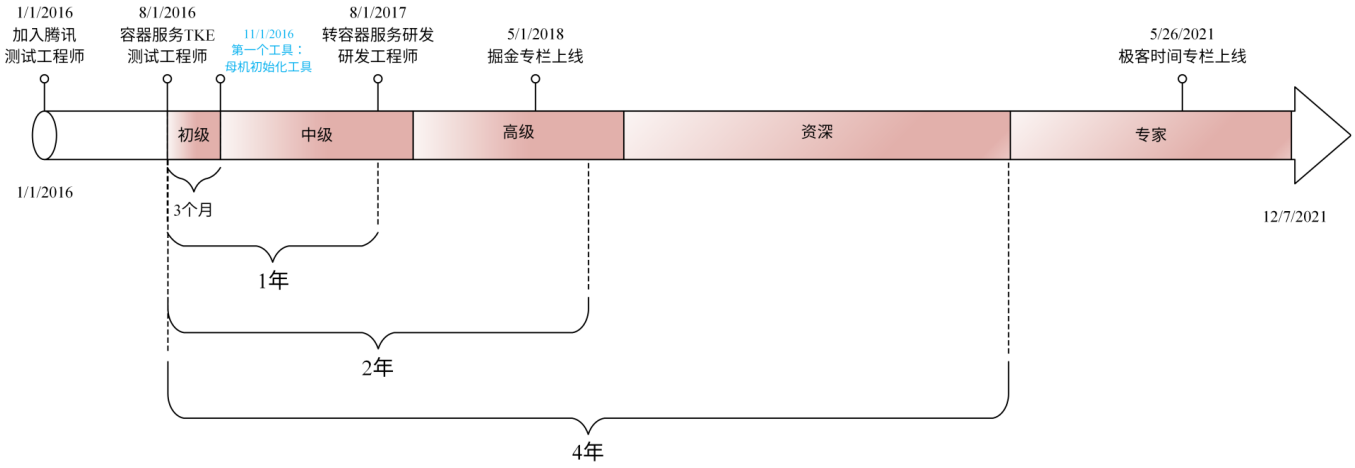
你可以从我的划分中看到，初级、中级、高级 Go 语言工程师的关注点主要还是**使用 Go 语言开发一个实现某种业务场景的应用**，但是资深和专家级别的 Go 语言工程师，除了要具有优秀的 Go 语言编程能力之外，还需要具备一些**架构能力**，他们的工作也不仅仅局限在开发一个 Go 应用，还需要兼具技术架构师的职责。

当你学习完《Tony Bai ·Go 语言第一课》时，你应该处在**初级**阶段，并具备向中级 Go 研发工程师晋级的能力，这个阶段也是所有 Go 语言程序员必经的一个阶段。接下来，你要做的是通过后期的打怪升级，晋级到资深，甚至专家级别。

那么如何晋级到资深 / 专家级别呢？其实没有标准答案，每个人都有自己的方法。但你可以借鉴别人的经验，来加快你的晋级速度。这也是今天这一讲想要达成的目标：**我希望通过分享我自己的晋级历程，帮助你快速晋级到资深 / 专家 Go 语言工程师。**

我的 Go 语言成长之路

这里我先简单介绍下我的 Go 语言进阶时间线。为了便于你理解，有些时间点，我做了四舍五入取整处理。整个时间线如下图所示：



我是在 2016 年 1 月加入到腾讯云的，当时的工作是虚拟化测试，对 Go 语言完全没有了解；

在 2016 年 8 月，我加入了腾讯云容器服务 TKE 团队，开始测试腾讯云容器服务 TKE 产品。因为 TKE 底层是用 Go 语言构建的，这之后我开始学习并使用 Go 语言；

2016 年 11 月，经过 3 个月的学习，我开发出了第一个小工具：母机初始化工具。母机初始化工具，其实就是在物理机上执行一系列的 Shell 命令，完成初始化工作。这个工具主要用到了 flag、log、os/exec、path/filepath、strings 等 Go 包，再加上使用 Go 语言控制语句实现的初始化逻辑；

在 2017 年的 8 月份，转岗到腾讯 IEG 团队，从事容器云平台的开发。也就是说，我用了一年时间完成了从测试到研发的角色转变；

在 2018 年 5 月份，撰写了掘金小册：《基于 Go 语言构建企业级的 RESTful API 服务》；

在 2021 年 5 月份，撰写了极客时间专栏：《Go 语言项目开发实战》。

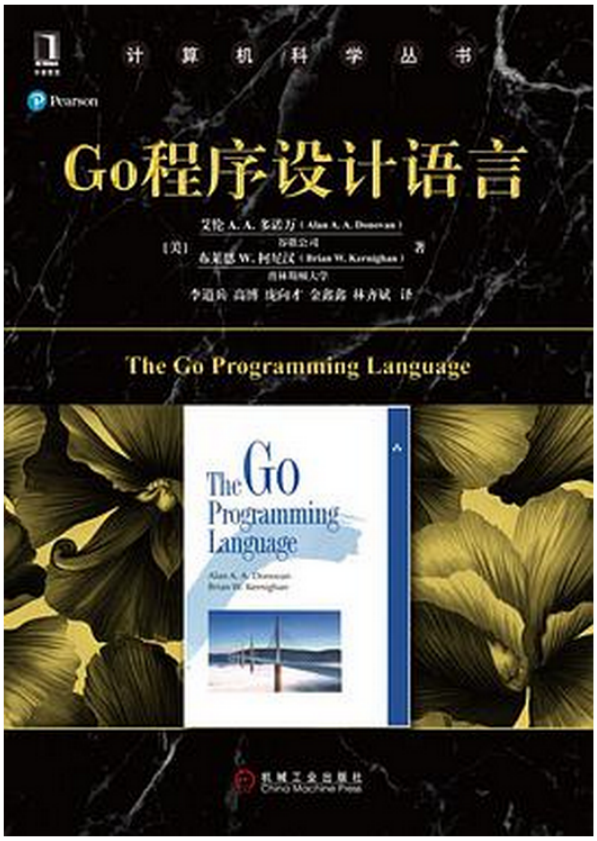
我当前还处在专家阶段。当然，专家也分等级，虽然这个阶段的等级不详，但学无止境，在 Go 语言这条道上，我永远是个学生，跟你一样，需要不断地学习、深造。

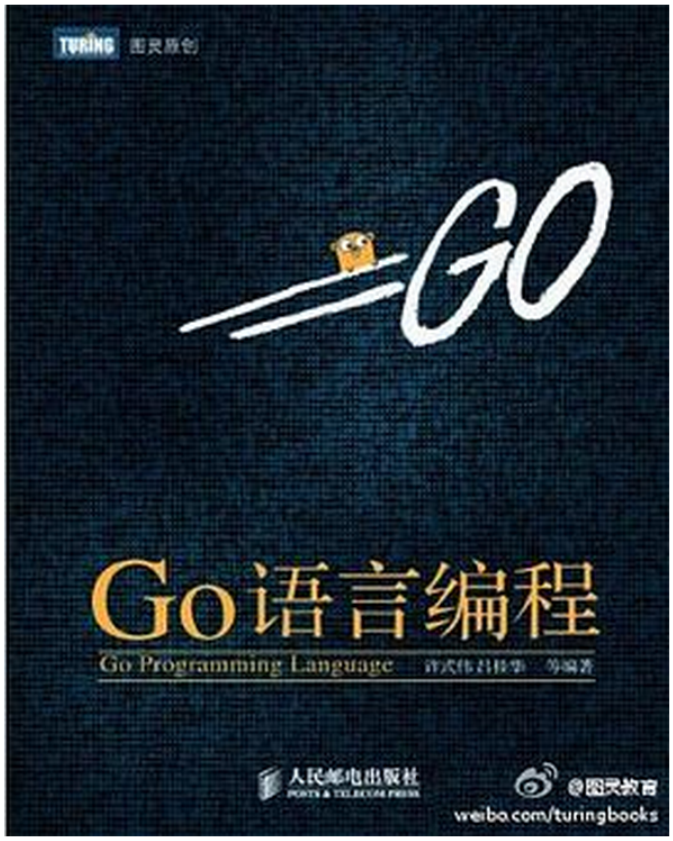
接下来，我会分享每个阶段我的经验、心得，希望这些分享能帮助你提高 Go 语言编程能力。

初级工程师阶段（2016-08-01 ~ 2016-11-01）

我是在 2016 年 8 月转入腾讯云容器服务 TKE 团队，测试 TKE 产品的，这也是我第一次接触到 Go 语言。因为 TKE 底层是用 Go 语言构建的，为了能够了解每次提测的代码变更内容，我需要能够读懂 Go 代码，这就需要我学习 Go 语言。

那么如何学习 Go 语言呢？在刚开始学习的阶段，最简单高效的方式就是看书。我当时买了两本书《Go 程序设计语言》和《Go 语言编程》（可惜当时没有 Tony Bai 老师的这个专栏）：





书本拿到手后，我先学习了《Go 程序设计语言》，过了一周后，我又通读了《Go 语言编程》。我一般学习一门编程语言，都会快速阅读两本经典的、讲基础语法的书。

这有两点原因。首先，学习一门课程，不能指望看一次就掌握。所以我会先认真阅读一遍书，不要求自己能记住 / 理解多少内容，但希望能够理解 / 掌握一些核心知识点，更重要的是能够了解书中有哪些知识点，这些知识点可以用在哪些地方。

另外，“工欲善其事，必先利其器”。学习一门语言最好的方式是多编码实操，但是在开始大规模实操之前，我们需要有一个踏实的基础。通读两本书，一方面我可以通过这两本书比较全面地掌握 Go 语言的基础语法，另一方面我也能让自己对核心知识点的记忆更加深刻。

在读完两本书，并且编写适量的简单 Go 代码段之后，我就具备了阅读其他项目代码的能力，这个时候就可以尝试向中级工程师晋升了。

接下来，我主要是通过**编码实战**加深对 Go 语法知识的理解和掌握。那么具体应该如何实战呢？在我看来应该以需求为驱动，找到一个合理的需求，然后实现它。需求来源于工作。这些需求可以是产品经理交给你的某一个具体的产品需求，也可以是能够帮助团队 / 自己提高工作效率的工具。总之，如果有明确的工作需求最好，如果没有明确的需求，我们就要创造需求。

在这个阶段，我会思考工作中的痛点、难点，并将它们转化成需求。比如，团队发布版本，每次都是人工发布，需要登陆到不同的服务器，部署不同的组件和配置。这样效率低不说，还容易因为人为失误造成现网故障。这时候，我们就可以将这些痛点抽象成一个需求：开发一个版本发布系统。

有了需求，接下来就要实现它，也就是进入到实战环节。那么如何实战呢？在我看来精髓在于两个字：“**抄**”和“**改**”。

现在，我们就基于前面“开发一个版本发布系统”的需求，分析看我们可以怎么通过“抄”和“改”来实现它。

如果自己从 0 开发出一套版本发布系统，工作量无疑是巨大的。而且，以我这个阶段的水平，即使花费了很多时间开发出一个版本发布系统，这个系统在功能和代码质量上也无法跟一些优秀的开源版本发布系统相比。

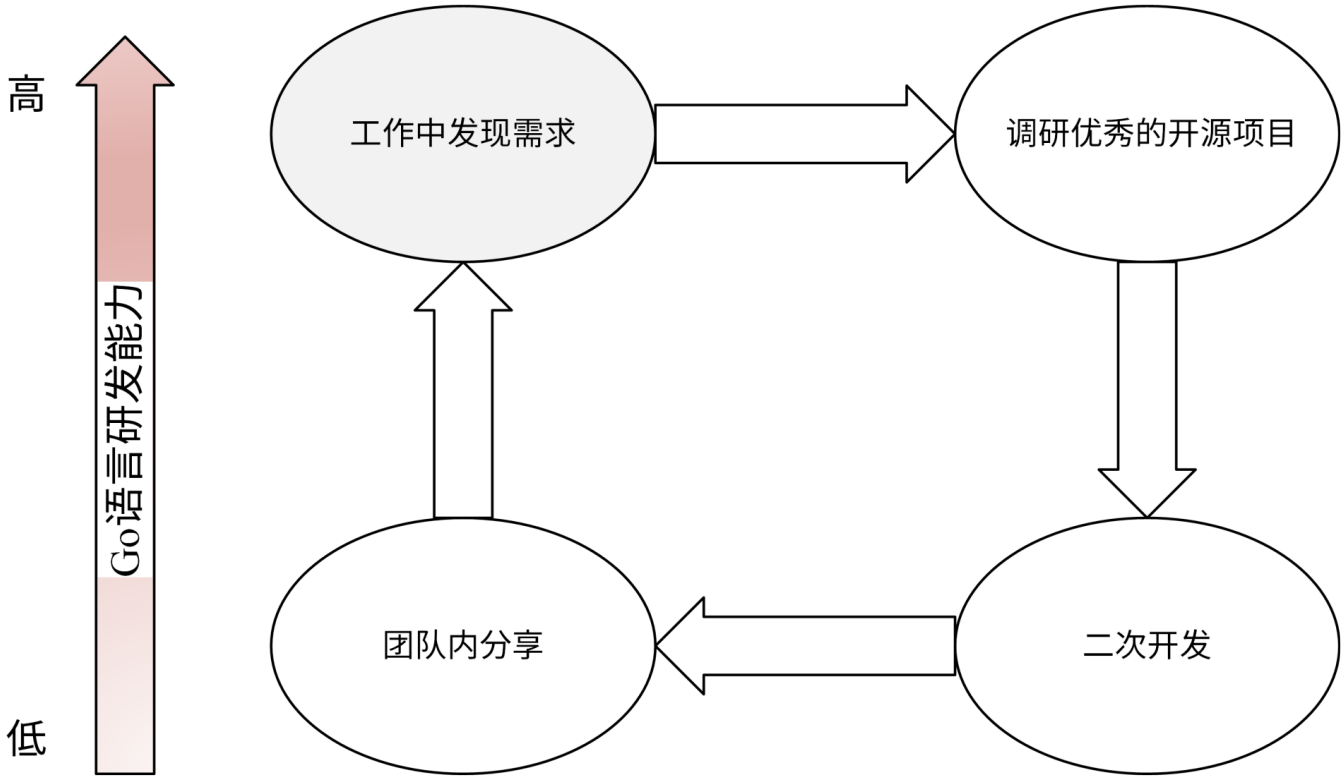
所以，这时候最好的方法就是在 GitHub 上找到一个优秀的版本发布系统，并基于这个系统进行二次开发。通过这种方式，我不仅学习到了一个优秀开源项目的设计和实现，还以最快的速度完成了版本发布系统的开发。

那么如何查找优秀的开源项目呢？我也有自己的一套方法，因为内容有点多，感兴趣的话，你可以参考我的专栏 结束语 | 如何让自己的 Go 研发之路走得更远？中的 “**问题二：如何查找优秀的开源项目？**” 部分。

当我完成了发布系统的二次开发之后，我还会在团队中进行分享，让自己的实战成果变成工作产出。至此，我也就完成了从初级工程师向中级工程师的晋升，进入中级工程师的阶段。

中级 / 高级工程师阶段（2016-11-01 ~ 2018-10-01）

中级 / 高级工程师阶段，其实就是不断地利用所学的 Go 基础知识，去编程实践。这个阶段提升 Go 研发能力的思路也跟前面是一样的：工作中发现需求 -> 调研优秀的开源项目 -> 二次开发 -> 团队内分享。通过这样一种循环过程，我们可以使自己的研发能力在循环中不断地提升：



在这样一种循环过程中，我通过不断地发现问题并解决问题，最终使自己能够熟练地掌握 Go 基础语法。不过，在这个阶段，我还做了一件事，就是刻意地减少对 Google/Baidu

的依赖，尝试自己编码解决问题、实现需求。在需要的时候，我也会使用 Go 的高级语法 channel、interface 等，结合面向对象编程的思想去开发项目或者改造开源项目。

在中级工程师阶段，我的工作还是测试，并没有来自于工作的 Go 研发需求，所以这个阶段为了学习 Go 语言，我虚构了很多“工作需求”，在完成这些虚构的“工作需求”的过程中，我的研发能力也有了非常大的提升，下面是我具体虚构的“工作需求”：

开发了母机初始化系统：

- 需求来源：因为我的日常工作需要初始化物理机，使其能够添加到腾讯云的资源池中。我之前的初始化工具，不具有重试、暂停、查看初始化详情、分布式控制等功能，使用起来很不方便，所以为了能够提高初始化效率和体验，我准备重新开发一个初始化系统；
- 调研开源项目：因为都是内部系统，我就直接基于所测试的容器服务底层组件来开发，这样一方面能够让我熟悉所测的项目源码，另一方面因为我之前已经读过它的源码了，二次开发起来难度也比较低；
- 效果：母机初始化系统开发完成之后，后续所有的母机初始化都是通过这个系统，大大提高了初始化效率和体验。最后，我将整个系统沉淀成文档，在团队内分享，推动其他同事使用这个系统，得到了领导的高度认可。通过这种方式，一方面工作上有了产出，
- 另一方面通过这个“虚构的项目”我也学到了很多实战技能。

开发了 HTTP 文件服务器：

- 需求来源：因为经常需要将同一个二进制文件部署到不同的机器上，为了便于分发文件，我开发了一个 HTTP 服务器；
- 调研项目：使用了开源的 gohttpserver；
- 效果：开发完成后，在团队中推广，有不少同事使用，显著提高了文件的分发效率。

命令行模板：

- 需求来源：因为经常需要编写一些命令行工具，所以我每次都要重复开发一些命令行工具的基础功能，例如命令行参数解析、子命令等。为了避免重复开发这些基础功能，提高工具开发效率和易用度，我开发了一个命令行框架，cmdctl；
- 调研项目：参考了 Kubernetes 的 kubectl命令行工具的实现；

- 效果：在工作中很多需要自动化的工作，都以命令行工具的形式，添加在了 cmdctl 命令框架中，大大提高了我的开发效率。

此外，我还研究学习了很多比较有趣的开源项目，你也可以参考一下，比如：

elvish：Go 语言编写的 Linux Shell；

machinery：Go 语言编写的分布式异步作业系统；

gopub：Go 语言编写的的版本发布系统；

crawlab：Go 语言编写的分布式爬虫管理平台；

还有不少其他好玩、有用的工具 / 项目。

在 2016 年 8 月 ~ 2017 年 8 月这一年间，我通读了两本经典的 Go 语言教材，并且调研、学习了大量的优秀项目，完成了多个虚构的“工作需求”，Go 研发技能有了非常大的提升，并具备了开发项目的能力。所以在 2017 年 8 月，我顺利转岗到了腾讯游戏部门，从事容器云平台的开发工作。

在开发容器云平台的 2 个月中（2017 年 8 月 ~ 2017 年 10 月），我的 Go 研发能力通过工作中 Go 语言相关项目的打磨、提升后，进入到了高级 Go 语言工程师的阶段。通过工作中的研发实战，也使我对自己的 Go 研发能力变得更加自信。

在学习其他开源项目的过程中，我积累了一些经验，也发现有些项目的构建思路比较清晰，代码质量比较高，有些项目代码质量、项目结构都很一般。并且，我还发现实际开发中，开发最多的项目就是 RESTful API 服务。于是，本着学习、总结、实战训练的目的，我在 2018 年 5 月撰写了掘金小册：《基于 Go 语言构建企业级的 RESTful API 服务》，这个小册会带着读者一步步构建 API 开发中的各个功能点，最终完成一个企业级的 API 服务器。

另外，在游戏部门工作期间，因为工作需要，我使用 Go 语言开发了微服务框架、API 网关、服务中心、CI/CD 等系统，这些工作内容帮助我学习了更多架构层面的知识，具有一定的架构能力后，我进入了资深工程师的阶段。

这里，我还想补充一点，我觉得初级、中级阶段可以通过自学来完成，但是想要进入高级、资深、专家级别更多的，或者必须通过工作中的 Go 开发实战才能进入。所以，如果

你想在 Go 研发之路上走的更深、更远，可以考虑在合适的时候切换到 Go 研发岗位上。

资深工程师阶段（2018-10-01 ~ 2020-08-01）

在资深工程师阶段，一方面我从 Go 语言项目开发层面继续打磨自己，另一方面，我还努力学习架构方面的知识。那么接下来，我分别分享下，这两个层面上我具体是如何做的。

Go 项目研发层面

在过往的 Go 语言学习过程中，我遇到了很多问题，主要分为以下 4 类问题：

知识盲区：Go 项目开发会涉及很多知识点，但自己对这些知识点却一无所知。想要学习，却发现网上很多文章结构混乱、讲解不透彻，搜索一遍优秀的文章，也要花费很多时间，劳神劳力；

学不到最佳实践，能力提升有限：网上的很多文章都会介绍 Go 项目的构建方法，但很多都不是最佳实践，学完之后不能在能力和认知上带来最佳提升，还要自己花时间整理学习，事倍功半；

不知道如何完整地开发一个 Go 项目：学了很多 Go 开发相关的知识点、构建方法，但都不体系、不全面、不深入。学完之后，自己并不能把它们有机结合成一个 Go 项目研发体系，真正开发的时候还是一团乱，效率也很低；

缺乏一线项目练手，很难检验学习效果：为了避免闭门造车，我们肯定想学习一线大厂的大型项目构建和研发经验，来检验自己的学习成果，但自己平时又很难接触到，没有这样的学习途径。

为了解决这些问题，并以此为驱动力和目标，我调研了 2000+ 开源项目、5000+ 国内外的技术文章，并根据这些调研，以最佳实践的方式开发了 Go 语言脚手架项目：iam。在调研、学习的过程中，我的 Go 研发能力，得到了大幅的提升。

在 2021 年 2 月，我尝试将过去学习过程中的一些心得、经验等知识沉淀成极客时间专栏《Go 语言项目开发实战》，希望以专栏的形式，分享给更多的 Go 研发工程师。经过近 4 个月的打磨，专栏于 2021 年 5 月上线。

如果你刚学完 Tony Bai 老师的《**Tony Bai • Go 语言第一课**》，接下来我建议你花点时间认真学习下《Go 语言项目开发实战》，这里面的内容使我突破到资深工程师。如果你

学习完之后，能够熟练的、独立开发类似的项目，并具有自己的理解，那么你的能力可能已经是或者接近资深 Go 语言工程师了。

架构层面

因为我本身就是做容器服务开发的，并且在 IEG 工作期间又从 0 到 1 自己构建了微服务、API 网关、CI/CD、服务中心等系统，所以在工作中，我的架构能力也得到了比较好的提升。

工作之外，我还通读了整个容器平台的核心代码，走读了容器平台所有组件的部署流程，了解了如何构建容器平台的监控告警（Prometheus）、日志（EFK）、容灾、DevOps（CODING）等核心能力的构建方式和部署方式。

同时，我在微信上又关注了一些云原生、架构相关的公众号，会利用闲碎时间阅读一些优质的公众号文章，再通过这些优质的文章，进一步丰富自己架构方面的知识。

关于如何提升架构能力，这里我有以下几点建议。

首先是学架构，先从当前业务开始。怎么开始呢？我们可以先了解当前业务的整体部署方式，最好是手动搭建一个小型的业务测试环境。在搭建过程中，你可以对业务的部署方式，有非常深刻的理解和掌握。

同时，我们可以走读所有的业务代码。可以先从核心代码开始读起，如果有时间，也可以通读当前业务所有组件的代码，最终你就能够知道整个系统是如何集成的，以及业务中每个功能是如何构建的。

接下来，学完当前业务架构，再学云原生架构。因为当前的软件架构都在朝着云原生架构的方向演进，云原生架构的基石是 Kubernetes。所以我建议你手动搭建一套原生的 Kubernetes 平台，然后在这个平台上部署一个小型的微服务系统，并构建这个微服务系统的日志、监控告警、调用链、服务发现等核心能力。

等集群、微服务系统部署好之后，你可以研究这个系统每一部分的部署和实现方式。任何的学习都离不开实战，这个 Kubernetes 平台以及其中的每一部分，都可以作为一个非常好的实战平台。

这里我推荐一个部署 Kubernetes 平台的教程：和我一步步部署 kubernetes 集群。这是一个傻瓜式的教程，跟着教程一步步操作，你就可以轻松地部署一套完整的 Kubernetes 集群。

此外，如果工作中有涉及到架构层面的工作内容，建议踊跃参与讨论、开发和实施。以工作为驱动，是学习架构最高效的，也是效果最好的方式，这种机会对于渴望提升自己架构能力的你，一定不要错过。

最后，你还可以关注一些优质的架构相关的公众号，利用闲碎时间，阅读优质的文章，补充自己架构方面的知识。

至于如何具体地提升自己的架构能力，因为内容比较多，这里我就不详细介绍了。如果你感兴趣，可以参考我的专栏：如何让自己的 Go 研发之路走得更远？ “**架构师阶段**”部分的内容。

专家工程师阶段（2020-08-01 ~ 至今）

这个阶段，我也是个学生，没有太多经验可分享。但我觉得还是可以使用之前的方式，不断从两个层面夯实自己的能力，也就是 Go 项目研发层面和架构层面。这个阶段，不仅要求你在这两个层面要走得更深、更远，还要求你能够兼具一个 Creator 的角色，能够从 0 到 1，构建满足业务需求的优秀软件系统，甚至能够独立开发一款备受欢迎的开源项目。

此外，这个阶段的你也要在团队中发光发热，兼具技术 Leader 的角色，把控技术方向、攻克技术难点，解决各种疑难杂症。

这个阶段是没有天花板的，所以这个阶段的你仍然要继续学习，通过不断地学习，成为一个越来越牛的技术达人。

Go 进阶之路心得分享

上面分享了我过往的 Go 进阶之路，希望对你能有所帮助。这里，我再总结下，在整个过程中，我认为比较重要的点。

第一点：尽快打怪升级。

程序员职业生涯短暂，竞争比较大，所以我们要通过努力，尽快实现 Go 语言开发能力的提升。想要加速提升能力，无外乎两个点：找对方法、多花时间。如果你刚毕业，或者还年轻，在保证身体健康的情况下，可以多熬熬夜，周末多加加班。未来的你，一定会感谢现在努力的自己。现在辛苦，换的是未来的轻松。现在小卷王，未来躺赢王。

第二点：找对方法很重要。

每个人都有自己的学习方法。我建议的方法是：工作中发现需求 -> 调研优秀的开源项目 -> 二次开发 -> 团队内分享。以工作需求为驱动，一方面可以让你有较强的学习动力、学习目标，另一方面可以使你在学习的过程中，也能在工作中有所产出，工作产出和学习两不误。基于优秀的开源项目二次开发，可以使你有动手实战的机会的同时，又可以学习到优秀开源项目的构建思路和构建方法。

第三点：学架构，先学习当前业务的架构，再学习云原生架构。

除了 Go 基本的项目开发能力之外，你平时还要注意积累自己的架构能力。积累架构能力最直接、高效的途径便是学习当前业务的架构，不仅要学习整个业务代码是如何实现的，还要学习整个软件系统是如何一步一步部署的。

此外，在云时代，我们还要学习云原生架构。学习云原生架构一个有效的方式是手动部署一个 Kubernetes 集群，并研究各部分是如何部署、甚至如何实现的。另外，提升架构能力最高效的途径是借助工作需求来提升，如果工作中有涉及到架构的工作任务，可以踊跃参与讨论、开发和实施。

最后，我还想补充介绍下我对**程序员职业生涯短暂**的理解。职业生涯短暂其实是一个伪命题，如果你够优秀，够努力，是可以一直在这个行业混的顺风顺水的。但是，我还是想说一些可能发生的残酷现实：程序员随着年龄的增长，工资越来越高，但精力、体力跟之前比也会有所下降，如果结婚生子之后，还要花费一部分的时间照顾家庭。

所以对于企业来说，毕业 3~5 年的程序员可能是性价比最高的，要时间有时间，要经验有经验，并且当前所积累的研发技能，已经能或者通过后期的学习能够满足公司业务开发需求了。如果公司遇到危机，需要裁员，可能会优先裁掉性价比低的那部分人。

那么，如何判断一个程序员的性价比呢？就是**你的能力要跑赢你当前的年龄和薪资**。想跑赢当前的年龄和薪资，需要你尽快地打怪练级，提升自己。

小结

今天的分享到这里就结束了。在这一讲中，我分享了我过往的 Go 进阶之路。内容很多，但在我看来核心就是三个点：

多花时间。建议你每天固定留出一些时间来提升自己的 Go 研发能力，比如下班后~ 晚上 12:00 前，每周也可以花费 1 天来提升自己；

找对方法。一个行之有效的方法是：在工作中发现需求 -> 调研优秀的开源项目 -> 二次开发 -> 团队内分享；

提升架构能力。提升架构能力可以先从当前业务开始，既要知道怎么部署，又要知道怎么实现。除了知道当前的业务架构之外，最好还能提升自己云原生架构的能力，提升云原生架构能力，可以通过搭建一个原生的 Kubernetes 平台，并部署一套小型的微服务，并为微服务构建日志、监控告警、调用链、服务中心等能力，然后再通过研究每一部分的部署和实现，补全自己的云原生架构能力。

如果你的工作中，有涉及到架构层面的工作需求，非常建议你踊跃参与讨论、开发和实施，因为这是提升架构能力最有效的方式。

思考题

1. 思考下，如何去调研一个优秀的开源项目，如果你有自己的方法，欢迎留言区留言讨论。
2. 思考下，你的工作中哪些地方可以抽象成一个有价值的需求，并用自己所学的 Go 语法知识，尝试实现它。

分享给需要的人，Ta订阅后你可得 20 元现金奖励

生成海报并分享

赞 7 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 28 | 接口：接口即契约

更多学习推荐

精选留言 (3)

写留言

lesserror
2021-12-20

学吧，太深了.....

3

无名无姓
2021-12-21

老师讲的真好

2021-12-20

输出倒逼输入，还是得多练习，才能不断进阶golang

作者回复: